Design and Analysis of a Programmable Single-Chip Architecture for DVB-T Base-Band Receiver

Chengzhi Pan, Nader Bagherzadeh, Amir Hosein Kamalizad, and Arezou Koohi,
all with ECE Department, University of California, Irvine, USA

Abstract

This work treats the design and analysis of a programmable (or reconfigurable) DSP-domain-specific architecture called MorphoSys, upon which world's first single-chip software solution for DVB-T base-band receiver can be implemented. Based on the first version of MorphoSys, many modifications have been made to improve greatly both computation power and data movement efficiency. Sequential codes and SIMD codes can be parallelized; temporal granularity adjustment boosts up performance up to 4 times; numerous different types of data movement can be accelerated 8 to 64 times faster than sequential movement. As a complicated (21GOPS) and typical communication system, DVB-T base-band receiver is designed with low performance loss and mapped onto MorphoSys architecture (>28GOPS). This solidly contributes to the software defined radio development.

1 Introduction

Digital Video Broadcasting--Terrestrial (DVB-T), the European Standard on digital TV radio by ETSI [1], is one of the most computation-intensive wireless communication systems. It adopts Coded Orthogonal Frequency Division Modulation (COFDM) scheme. The OFDM symbol length is as large as 2K/8K with the highest data rate of 31.67Mbps. The base-band receiver is composed of inner receiver (OFDM demodulator), outer receiver (64-state Viterbi decoder and Reed-Solomon decoder), and source decoder (MPEG-2 decoder). How to implement this computational-intensive receiver onto a single programmable device is an extremely challenging task. A multi-chip quasi-software solution exists [2], requiring no less than 10 chips. This paper proposes a pure-software single-chip solution based on a reconfigurable architecture called MorphoSys. Firstly, DVB-T receiver system design procedure is described and a fixed-point simulation result conforming to the protocol is achieved. Then, we introduce the architecture and programming model of MorphoSys processor. At last, algorithm mapping procedure and the results will demonstrate the architecture having orders of magnitude higher computation capability than other commercial DSP chips.

2 DVB-T system design

2.1 DVB-T base-band receiver

Base-band digital signal processing (DSP) subsystem of DVB-T receiver receives digital signal from ADC, then completes tasks such as OFDM demodulation, channel decoding, and MPEG-2 decoding (diagram shown in Fig.1).

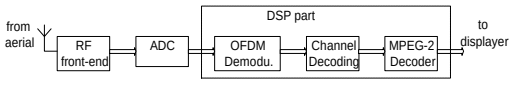

Fig. 1 Diagram of DVB-T receiver

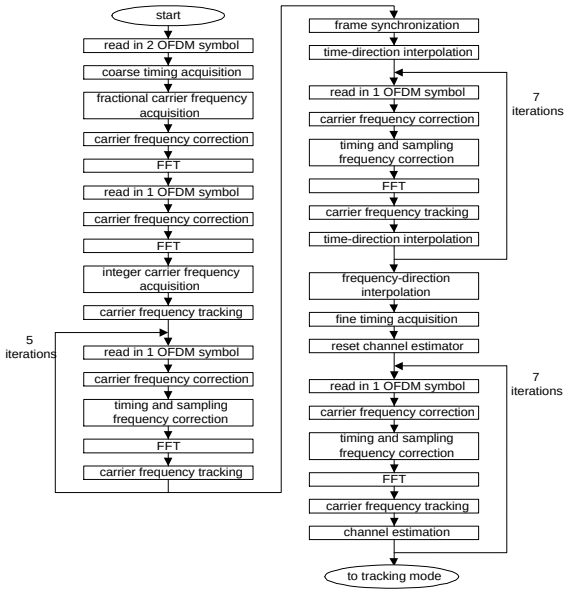

Fig. 2 Task flowchart in acquisition mode (10 diff. blocks)

2.2 Floating point system design and simulation

We choose a computation-intensive system configuration of DVB-T as our design target: Single Frequency Network (SFN), 8MHz channels, 8K mode, 64-QAM, ¼ Guard interval, ½ convolutional code rate, which has "a good compromise between bandwidth

efficiency and robustness"[3]. The floating point system design is divided into 3 parts:

First is the algorithm design. There are two modes of the receiver: acquisition mode and tracking mode. As soon as the receiver is turned on, it goes into acquisition mode (Fig.2). After 22 OFDM symbols (24.6ms), timing and frequency acquisition is achieved and the receiver turns into tracking mode (Fig.3). Most of the algorithms for above blocks are similar with those in references [3-5], etc.

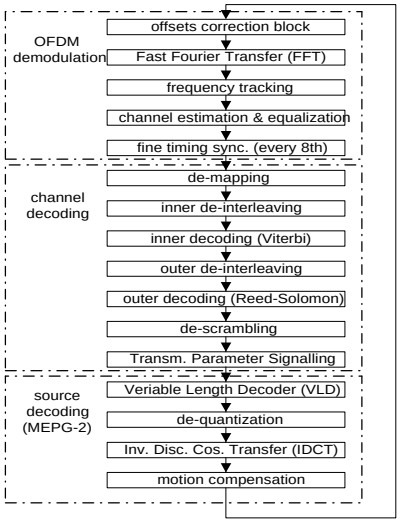

Fig. 3 Task flowchart in tracking mode

Second part is designing an accurate and fast channel simulator. As a widely used accurate model, a multi-path Rayleigh fading, wide-sense-stationary uncorrelated-scattering (WSSUS) channel model is adopted. Then a fast WSSUS channel simulator is written based on the one in reference [6]. As the most difficult scenario, "typical case for hilly terrain (HTx)" specified in 3GPP standard [7] is used. SFN is modeled as HT1 followed by HT2 with a delay of half guard interval.

Parameter optimization, as the third part, is to minimize the computation requirement while still meeting the Quasi-Error-Free requirement (BER<$2*10^{-4}$) after Viterbi Decoding in tracking mode. Following is the main result of this SNR budgeting process. (1) Viterbi decoder: since soft decision Viterbi is 3.5dB better than hard decision Viterbi (Fig. 4), soft decision algorithm is adopted and the input SNR needed is 13.2dB. (2) Channel estimator: A robust channel estimator using Weiner filter with three sets of coefficients (12 each) is optimized. The estimator gain can be 4~5 dB. (3) Frequency tracking block: a 25-tap 4-fold interpolator followed by a linear interpolator is used to compensate sampling frequency offset. There is no adjustable parameter in outer receiver and source decoder. Simulation shows that in order to reach QEF, the input channel SNR is 20dB for HT1 channel, and 25dB for SFN (Fig. 5).

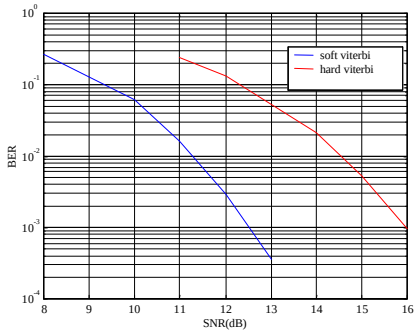

Fig. 4 Viterbi performance comparison

2.3 Fixed point system design and simulation (SQNR budgeting)

Fixed-point system design on a digital signal processor is a process of trade-offs between dynamic range and quantization error. Careful optimization of the fixed-point system on the 16/32-bit MorphoSys architecture [8-9] results in a negligible performance degradation (see Fig. 5). Operations such as cosine, sine, division, Galois Field multiplication, are efficiently implemented using Look-Up-Tables (LUTs) or small LUTs followed by linear interpolators.

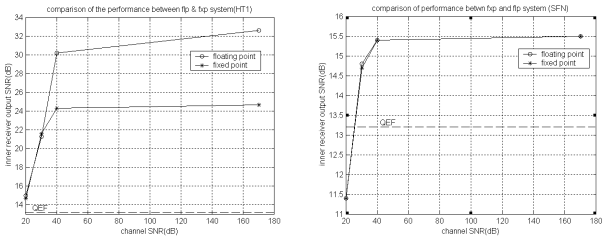

Fig. 5 Inner receiver performance (HT1&SFN channel)

3 MorphoSys Architecture

3.1 MorphoSys architecture introduction

MorphoSys is a domain-specific reconfigurable architecture targeting on data-parallel and computation-intensive applications, such as 3G/4G wireless communications, multimedia processing, high quality graphics, etc. [8-9]. It can achieve high performance near ASICs while having great flexibility as DSPs. Moreover, its power consumption is lower than DSPs' and FPGAs'. The system consists of three main subsystems: a core processor called TinyRISC processor, an array of 64 Reconfigurable Cells (RCs) organized in SIMD fashion, and a special data movement unit called Frame Buffer

(FB). The programming model is simple. TinyRISC takes charge of the whole Data Control Flow (CDF); RC array and Frame Buffer are only triggered by TinyRISC and executing on their own configurations (called context) continuously for a given number of cycles specified by TinyRISC.

Based on the design of the first version of the MorphoSys architecture called M1 (see Fig. 6) and numerous mapping experiences thereafter, many modifications have been made to design an improved version of the architecture called M2 that is much faster, more memory efficient, and less power dissipating. Following is the description of these modifications.

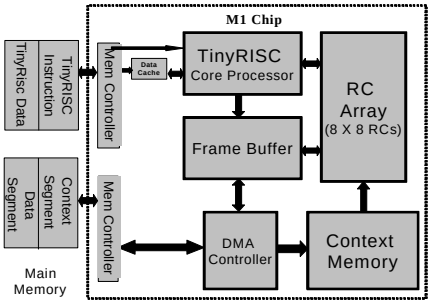

Fig. 6 MorphoSys architectural model (M1)

3.2 TinyRISC modifications

3.2.1 Decoupled execution of TinyRISC and RC array In M1, TinyRISC has to issue one instruction to specify the Context Memory address for each cycle when RC array is executing context. Through application mapping, we find out that most of the contexts are executed continuously for a certain number of cycles up to 150. Hence, in M2, TinyRISC is running a special sequencing thread, which specifies the starting address of context memory and the number of contexts to be executed, i.e., starts a piece of block codes on RC array. After that, TinyRISC turns to do other tasks (threads) unrelated to RC array, while RC array is increasing the context memory address automatically. After the block code on RC array ends, TinyRISC again automatically turns back to the sequencing thread (Fig. 7).

There are great benefits for this decoupled execution of TinyRISC and RC array. Firstly, this will make possible the overlap of execution of parallel codes on RC array and sequential codes on TinyRISC. In large systems, there are often some irregular computations and/or naturally sequential tasks, e.g., Variable Length Decoding in MPEG-2 decoder, Trace-Back in Viterbi Decoding, etc., which can hardly be parallelized onto RC array. They have to be done on TinyRISC sequentially. Overlapping them with other parallel tasks on RC array by high-level pipelining will greatly reduce the total execution time. Besides, this decoupling feature can also reduce the TinyRISC code size sometimes in orders of magnitude. Furthermore, it also saves power by reducing instructions issued.

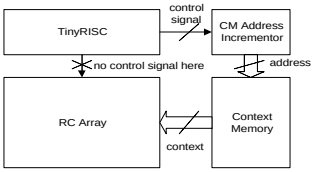

Fig. 7 Decoupled execution of TRISC and RC array

3.2.2 Subroutine support In order to support modular programmability for mapping large system, a low-overhead (1 cycle) subroutine call/return TinyRISC-instruction pair is added. The PC stack depth of 10 is proved enough.

3.2.3 Zero-overhead loop In communications and DSP applications, there are often many loops with large number of iterations, and each iteration contains only few instructions. E.g., for data movement there is often only one instruction in each iteration. However, loop overhead is 3 cycles. A zero-overhead loop TinyRISC-instruction can boost the performance by 4 times.

3.2.4 Other modifications Instruction set is also modified according to above innovations and following ones as well. Other new instructions, such as multiplication, sleep, etc., give the processor a better trade-off between area, speed, and power. Interrupt unit is also added to deal with interrupts from timer, DMAs, and off-chip requests.

3.3 RC array modifications

3.3.1 Temporal granularity: key of high-speed architecture As the operation profiling of the whole system indicates, Multiplication & Accumulation (MAC) operations occupy only less than 10% of the total cycle number, while MAC unit contributes 2/3 to the critical-path delay. This causes a great mismatch between the temporal granularity of the architecture (RC array) and that of the algorithm. Then comes the solution: pipeline the MAC unit in 3 stages and shorten the critical path. Circuit simulation shows that the critical path is made up of the following parts with conservatively estimated delay (Fig. 8): (1) Register setup time and hold time, including clock skew: 0.2 ns; (2) Inter-RC connections: 0.5~1 ns; (3) Input MUX (overlapped with inter-RC connections): 0.3 ns; (4) ALU: 1.2 ns; (5) Output MUX: 0.3 ns. After optimization, Synopsis post-synthesis simulation using 0.13μm technology typical library shows that the critical-path delay excluding inter-RC-connection delay is 1.3ns. Thus the total critical-path delay turns out to be 1.8~2.3ns (556~435MHz).

Compared with the former architecture of 8ns delay (125MHz), the performance without MAC operations is boosted up by 4 times or so. For applications with n percent of non-pipelined MAC operations, the performance will drop by 2*n percent. For DVB-T receiver, n is less than 10. As a result, the speed of M2 is beyond what the DVB-T receiver needs as 406MHz (shown in Table 1).

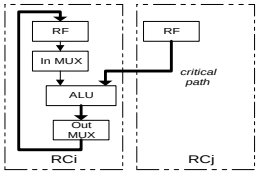

Fig.8 Critical Path

3.3.2 Inner-RC memory Inner-RC Memory (4KB SRAM in each RC) is the main storage media in the architecture. According to the SIMD nature of the application domain, data assessments are often local, i.e., within each RC. Thus, global data movement can be minimized by using Inner-RC Memory. Moreover, this memory is also shared with TinyRISC to allow easier data communication between TinyRISC and RC array.

3.3.3 Enhanced inter-RC connection For a large system, there is always some residual data movement between RCs even after careful minimization, such as butterflies in FFT, metrics movement in Viterbi Decoding, and rearrangement of edge data for FIR filters, etc. Fortunately, these are regular data movement and can be parallelized using enhanced inter-RC connection networks, which has all-to-all connections for each column and each row.

3.3.4 RC active mask It is not always the case that all the RCs are busy. Sometimes it is even mandatory to idle some RCs, e.g., on metrics movement in Viterbi Decoding. RC-activeness-selecting masks are introduced to solve this problem. This also saves the size of contexts.

3.3.5 Context set modification Based on the profiling of mapping results on communications and DSP systems, the context set is optimized to match the domain requirements tighter while still persisting enough generality. For example, a bit copy instruction is added to meet the requirement of many bit-wise algorithms.

3.4 FB modifications

3.4.1 Fast connection between FB and RC array FB is constructed in 64 banks corresponding to 64 RCs, so that data communication between FB and RC array will be as fast as 64 Words (128 Bytes) per cycle. FB is a two-port memory, which can overlap the time when data are buffered from External Memory to Inner-RC Memory through FB. Since it is often needed to copy one Look-Up-Table into 64 RCs, FB also has the data broadcasting capability.

3.4.2 Fast continuous data movement within FB In large systems, data output from one code block often needs to be rearranged before it can be used as input to the next code block. This data rearrangement is a bottleneck of many parallel architectures. For example, the simple zero-padding operation of a vector in a SIMD architecture will become very difficult. However, most of these rearrangements are either regular movement with parallelism on RC arrays or continuous data movement, or combination of both. In DVB-T receiver, except pseudo-random de-interleaver, all other data movement falls into these three categories. For movement with parallelism on RC arrays, inter-RC connection can deal with it well. For continuous data movement, a special mechanism in FB is designed to accelerate the speed up to 32 Bytes per cycle. The hardware costs are 32 16-bit shifters and more sense amplifiers in SRAM.

3.4.3 Shift between two addressing order Due to the bank structure of FB, FB can be regarded as a two-dimensional memory cell (16-bit) array. There are two ways to order the address: one is addressing crossing banks; another is addressing along banks (Fig. 9). Sometime it is necessary to shift data between these two orders. E.g., the order of data after FFT will change from the second order to the first order. Hence, a special 8K-bit block of FB is designed. The hardware innovation is to route two sets of bit-lines and word-lines in orthogonal directions.

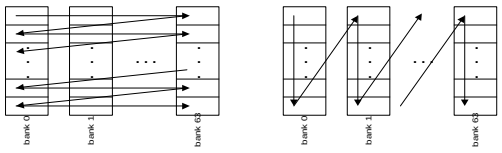

Fig. 9 Two addressing order of FB

3.4.4 Direct Memory Movement (DMM) Except regular types of movement above, there still may be some naturally irregular one, e.g., Pseudo-Random Deinterleaving, whose very purpose is to disorder the data in a pseudo-random manner. Sequential solution on TinyRISC is costly in time, about 64K cycles. Thus, a DMM unit is introduced to remove the overhead and finally reduce the cycle number to 24K. DMM read data and destination address in one cycle and write data in another cycle (Fig. 10). An additional benefit of DMM unit is to free TinyRISC as well.

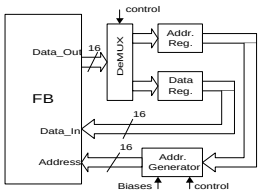

Fig. 10 Diagram of DMM

Thus, all these modifications result in more TinyRISC computation power, less CDF control overhead, smaller code size, maximized RC speed, reduced data movement overhead, numerous different types of data movement accelerated 8 to 64 times faster than sequential movement, and lower power consumption with small hardware cost.

4 Mapping DVB-T onto MorphoSys Architecture

4.1 Mapping methodology

Since the critical path of the system lies in tracking mode, we only consider mapping of tracking mode. Thanks to the simple programming model the architecture adopts, the mapping methodology is not much more complex than general-purpose processor. The whole CDF is controlled by TinyRISC. RC array is simply executing block code without branches, behaving like a multi-cycle function unit. FB can be viewed as another function unit.

TinyRISC code is written in assembly language, which has some high-level language features such as loop, subroutine, etc., to ease the programming. Multi-file modular programming is also supported.

The key of a good mapping is to extract maximum parallelism from algorithms. There are two types of parallelism suited for the architecture: (1) SIMD within RC array, and (2) MPMD between TinyRISC and RC array. In DVB-T, in most cases SIMD parallelism is extractable, e.g., FIR filters for Channel Estimation, Butterflies in FFT, Metrics Computation in Viterbi Decoding, etc.. Regular data movements between RCs, e.g. those in FFT, Viterbi Decoding, and Outer De-Interleaving, can also be implemented in parallel. MPMD parallelism is achieved by overlapping sequential codes on TinyRISC and parallel codes on RC array. Besides, DMM also provides another MPMD between FB and other components.

Other patterns of data movement, which is essential for data-intensive parallel computation system, is also well addressed by enhanced FB and inter-RC connection.

4.2 Mapping illustrations

Due to the length limitation of the paper, here we can only illustrate the whole system mapping with two typical examples.

The first example is Channel Estimation and Pseudo-Random Deinterleaving. Since Deinterleaving is dealt with sequentially by DMM (see Section 3.4.4), more cycles can be saved by overlapping it with other block running on RC array and/or TinyRISC, i.e. Channel Estimation. The solution is to pipeline these blocks and utilize the MPMD parallelism between FB and other components (Fig. 11). Thus, nearly 99% of the cycles for Channel Estimation block are hidden.

Another example is Viterbi Decoding, the most computation-intensive block. The same high-level pipeline technique is adopted to hide the sequential

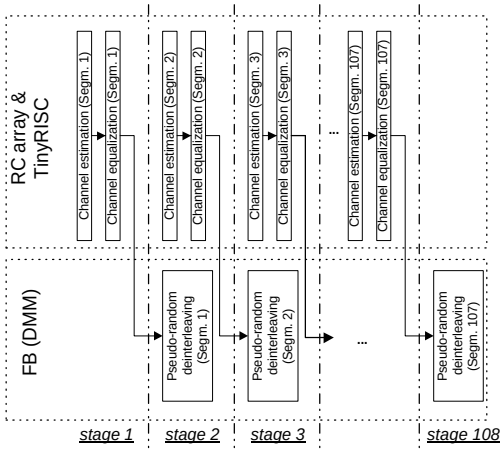

Fig. 11 Pipeline of Channel estimation & equalization and Pseudo-random deinterleaver

Trace-Back block in TinyRISC with Metrics Computation block for the next segment of data in RC array (Fig. 12). This MPMD parallelism between TinyRISC and RC array totally hides the Trace-Back block, resulting in around 30% reduction of the cycle number. High speed of 9 cycles per decoded bit, corresponding to 47~60Mbps/chip under the current technology, is near the ASIC performance, while no special Add-Compare-Selection instruction is added, unlike other DSPs do.

4.3 Mapping result

Table 1 gives out the mapping result estimate of the whole system. We can draw some conclusions from that: (1) It is world's first single-chip software solution of DVB-T base-band receiver based on the current technology. (2) Code size is small. (3) Data memory needed is mostly required by the protocol; extra temporary memory needed is relatively small and can be further reduced.

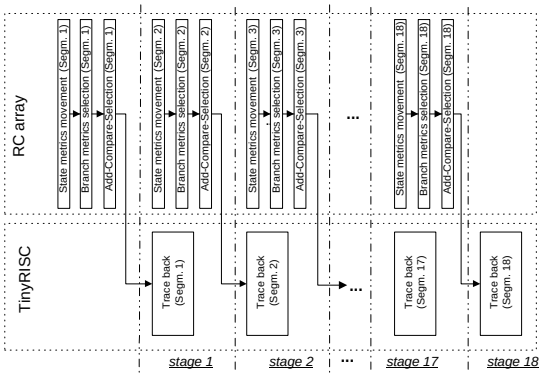

Fig. 12 Pipeline implementation of Viterbi decoder

Table 1 Mapping result estimate

Units	Cycle #[1] (K)	Permanent storage[2] (KB)	Tempo-rary storage (KB)	Context size (KB)	TRISC inst. Size (KB)
Buffer add. calculation	1	32	0.0	0.0	0.2
Carrier freq. correction	11	32	0.1	0.3	0.1
Other offset correction	17	0.0	34	0.3	0.1
FFT	24	0.0	8	0.9	1.2
Freq. offset track	1	0.0	0.5	0.1	0.4
Fine timing sync.	1	0.0	16	0.1	1.3
Channel estimation[3]	17	0.0	3	0.1	0.2
Channel equalization[3]	5	96	8	0.1	0.1
Random deinterleaving[3]	24	0.0	13	0.0	0.1
Demapping[4]	4	0.0	8	0.2	0.1
Bit-wise deinterleaving[4]	7	0.0	0.0	1.0	0.3
Viterbi decoder[4]	175	0.0	8	3	0.5
Outer deinterleaving	5	0.0	5	0.1	0.1
Reed-Solomon decoder	5	0.0	48	3.2	0.4
Descrambler	1	0.0	5	0.0	0.0
MPEG-2 decoder[5]	179	0.0[6]	64	4.0	7.0
Total or maximum[7]	455	160	64	13.4	12.1

1 The OFDM symbol duration is 1.12ms.
2 The permanent memory doesn't count in what has been required by former units.
3 These units are pipelined and overlapped so that the total cycle# is 24K.
4 These three units are pipelined to reduce the temporary storage.
5 MPEG-2 decoder (MP@ML) mapping estimate is mainly made based on the previous mapping of parallel kernels in [8], except for that sequential code on TinyRISC for VLD is estimated according to many references such as [10]. Noted that overlapping of the sequential VLD with parallel code for channel Demodulation&Decoding subsystem could further reduce the cycle# much lower. Only Video decoding is considered.
6 Reference Pictures are stored in External Memory.
7 It is assumed that the loading of all the temporary data, context, and TRISC code from External Memory is 100% overlapped with the execution of other units. This assumption will cause cycle# estimation error within 30K, most probably.

5 Conclusions

A single-chip software solution of DVB-T base-band receiver on MorphoSys reconfigurable architecture is presented. Compared with other existing DSP solutions, there is a significant improvement of computation power per chip, even after considering the factor of technology improvement. This improvement comes from the high parallelism of both computation and data movement with very small overhead, i.e., excellent match between the architecture and application domain. Due to the huge scale and typical-ness of the system, the architecture has plenty of generality in domain of DSP, Communications, and Multimedia. No special hardware is tailored to any specific function. Further research directions include fine-grain algorithms match, parallel compiler design, lower power design and layout design, etc.

Acknowledgements:

This work was sponsored by DARPA (DoD) under contract F-33615-97-C-1126, the National Science Foundation (NSF) under grant CCR-0083080, and State of California CoRe funded research in cooperation with Broadcom Corporation.

References:

[1]: "Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television", ETSI EN 300 744 v1.4.1, January 2001.
[2]: F. Frescura, etc., "DSP based OFDM demodulator and equalizer for professional DVB-T receivers", IEEE Trans. Broadcasting, vol. 45, No. 3, pp. 323-332, Sept. 1999.
[3]: M. Speth, S. Fechtel, G. Fock, and H. Meyr, "Optimum Receiver Design for OFDM-Based Broadband Transmission—Part II: A Case Study", IEEE Trans. Commun., vol. 49, No. 4, pp. 571-578, Apr. 2001.
[4]: P. Hoeher, S. Kaiser, and P. Robertson, "Two-dimensional pilot-symbol-aided channel estimation by Weiner filtering", IEEE Conference on Acoustics, Speech, and Signal Processing, vol. 3, pp. 1845-1848, 1997.
[5]: J. Hsu and C. Wang, "An Area-efficient Pipelined VLSI Architecture for Decoding of Reed-Solomon Codes Based on a Time-Domain Algorithm", IEEE Trans. Circuits and Systems for Video Technology, vol. 7, No. 6, pp. 864-871, Dec. 1997.
[6]: A. Anastasopoulos and K. Chugg, "An efficient method for simulation of frequency selective isotropic Rayleigh fading", IEEE 47th Vehicular Technology Conference, vol.3, pp. 2084-2088, May 1997.
[7]: "Digital cellular telecommunications system (Phase 2+); Radio transmission and reception", ETSI TS 100 910 v8.9.0, Apr. 2001.
[8]: H. Singh, etc., "MorphoSys: An Integrated Reconfigurable System for Data-Parallel and Computation –Intensive Applications", IEEE Trans. Computers, vol 49, No. 5, pp. 465-481, May 2000.
[9]: M. Lee, etc., "Design and Implementation of the MorphoSys Reconfigurable Computing Processor", Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology, vol.24, No.2-3, Kluwer Academic Publishers, pp. 147-164, Mar 2000.
[10]: Ishii, D., etc., "Parallel variable length decoding with inverse quantization for software MPEG-2 decoders", IEEE Workshop on Signal Processing Systems, pp. 500-509, Nov 1997.